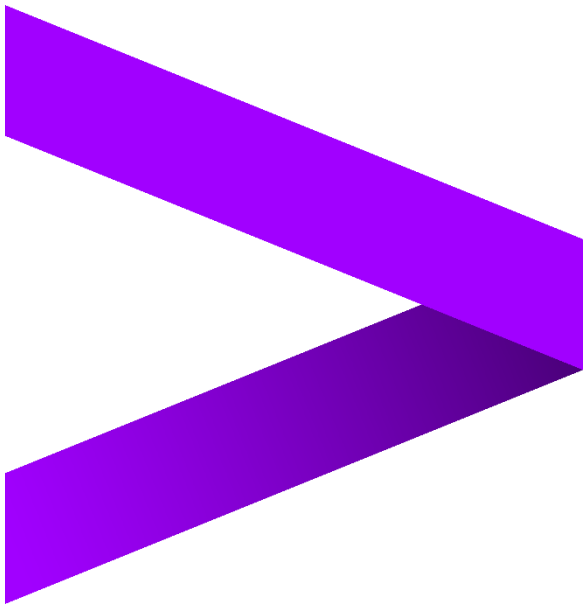
accenture

Learning and Knowledge Management

Mastering Ansible

Working with Ansible-Vault



Version	Revision Date	Description	Author(s)	Reviewed by	Approved by
V1.0	14-09-2020	Initial Version	Ranjith Kumar Thirumalai Ramesh	Rathnajyothi Perumalsamy Vishnu Kallimakula	CPCL Sreenivasa Rao

Index

Exercise: Working with Ansible-Vault

Prerequisite

Scenario: *In this activity, we will cover how to secure the playbooks with Ansible Vault.*

Sub Activity 1: How to use ansible-vault command to encrypt a file, decrypt a file, change the encryption password of already encrypted file, view and edit encrypted file and delete the encrypted file.

Step 1	Create a directory vault-demos .
	
	mkdir vault-demos && cd vault-demos
	
	Create a file called **playbook.yml** file with encryption using below command –
	
	ansible-vault create playbook.yml
	
	It will prompt for the password. Give some password. You must remember the password. Otherwise, you cannot open/edit/delete the file.
	
	`[root@localhost vault-demos]# ansible-vault create playbook.yml`
	`New Vault password:`
	`Confirm New Vault password:`
	`[root@localhost vault-demos]#`
	
	It will open **vi editor** → press '**i**' key on the keyboard to change to insert mode → copy the text given in **playbook.yml** file supplied, and paste into the vi editor opened → press '**Esc**' key on the keyboard to come out of insert mode → press '**:wq**' keys to exit from vi editor by saving the content you have pasted into **playbook.yml** file
	
	To test if the **playbook.yml** file is encrypted, try to see the file content by executing the below command –
	
	cat playbook.yml
	
	You can see the output as shown below –

```
[root@localhost vault-demos]# cat playbook.yml
$ANSIBLE_VAULT;1.1;AES256
31356633343937333833383538333464356536323234623534366231633337333931326432633737
62393639336466373731336363386261383638383764653806266626661623335663632376376664
63313637666162346564643162316432656262323930353930303230623038376165666633626531
38303565393161303306a313764623866363331333664613064643331323937613865343737383538
63353134363161333936343937333635313630663835303066346239633261616265656336323434
36653239306438396332316336333832323439303261313162643734303633373737383631353939
34383961623232393431653964303566613363633739303262623830336438616365623263663438
33353932643939356531386139316633353134663039306237666639393932663431613561373930
65313162383630333030363831623666363139313332626265396162356562333365646535353331
65323566623664353763613139616430303663336636323232313462633862626335313961386232
37613633393737393564646630623936356439613636613163333963326462323465356664643561
61326531626337616361306166313430393239373132613635353833366332303164326532663837
63646430643236666639356130663164346435536363166626366303336636643035
[root@localhost vault-demos]#
```

Step 2

To see the content of the encrypted file, execute the below command –

ansible-vault view playbook.yml

It will prompt for password you have given while encrypted the file. Once you give the same, you can see the file content. The output of the above command is as shown below –

```
[root@localhost vault-demos]# ansible-vault view playbook.yml
Vault password:
---
- hosts: target
  become: yes
  gather_facts: false

  tasks:
  - name: Create a file with some content
    copy: "dest=/tmp/file1.txt content='Hi, Hello, How are you?'"
[root@localhost vault-demos]#
```

Note: **ansible-vault view** command helps you only to view the file content

Step 3

To edit the **playbook.yml** file, you can use the below command –

ansible-vault edit playbook.yml

It will prompt for password you have given while encrypted the file. Once you give the same, you can see the file content in vi editor.

If you want to modify the file –

press '**i**' key on the keyboard to change to insert mode → modify the file as you want → press '**Esc**' key on the keyboard to come out of insert mode → press '**:wq**' keys to exit from vi editor by saving the content you have modified into **playbook.yml** file

You can see the output as shown in the below screen –

```
[root@localhost vault-demos]# ansible-vault edit playbook.yml
Vault password:
[root@localhost vault-demos]#
```

Step 4

To encrypt file already existing, you may follow the below procedure –

a) Create a file (may be with the name **helloworld.txt**) with some content using the below command –

echo "Hi, I am demonstrating the ansible-vault encrypt command." > **helloworld.txt**

```
[root@localhost vault-demos]# echo "Hi, I am demonstrating the ansible-vault encrypt command." > helloworld.txt
[root@localhost vault-demos]#
```

b) Execute the below command to confirm if the file created with the content above

cat helloworld.txt

```
[root@localhost vault-demos]# cat helloworld.txt
Hi, I am demonstrating the ansible-vault encrypt command.
[root@localhost vault-demos]#
```

c) Execute the below command to encrypt the file helloworld.txt file –

ansible-vault encrypt helloworld.txt

It will prompt for the password. Give some password. You must remember the password. Otherwise, you cannot open/edit/delete the file.

```
[root@localhost vault-demos]# ansible-vault encrypt helloworld.txt
New Vault password:
Confirm New Vault password:
Encryption successful
[root@localhost vault-demos]#
```

d) Check if the file is encrypted using the cat command as we did earlier –

cat helloworld.txt

```
[root@localhost vault-demos]# cat helloworld.txt
$ANSIBLE_VAULT;1.1;AES256
61363661343064653534666236393038626562656664363132313763343062616632383138386430
36323464366231373535656436616232643766356437323 70a31643934386461356562316366333 1
66633063616537303238636434306231613161336436386365376239326637333037303466343430
30353738623161386 10a6134333364393765616664393130373238393532636632613265383537 32
33636135643036646361653866656337306236653938383361313563623465393335626135633139
38303338636462643162376536343732353261396332633532643037663363633564306130623465
38353432366362333265663135366335666 6
```

e) If you use **ansible-vault view** or **ansible-vault edit** commands, you can see the file content –

```
[root@localhost vault-demos]# ansible-vault view helloworld.txt
Vault password:
Hi, I am demonstrating the ansible-vault encrypt command.
[root@localhost vault-demos]#
```

Step 5

If you want to change the encryption password given while creation/encryption of file, you should use **ansible-vault rekey** command as shown below –

ansible-vault rekey helloworld.txt

The above command will help you to change the vault password for the already encrypted file **helloworld.txt**

The output you can see below –

```
[root@localhost vault-demos]# ansible-vault rekey helloworld.txt
Vault password:
New Vault password:
Confirm New Vault password:
Rekey successful
[root@localhost vault-demos]#
```

Step 6

If you want to unencrypt/decrypt the encrypted file, use the command **ansible-vault decrypt.** Let us understand the **ansible-vault decrypt** command by decrypting the file helloworld.txt –

a) Execute the cat command to see if the file helloworld.txt is still encrypted –

cat helloworld.txt

```
[root@localhost vault-demos]# cat helloworld.txt
$ANSIBLE_VAULT;1.1;AES256
61363661343064653534666236393038626562656664363132313763343062616632383138386430
36323464366231373535656436616232643766356437323 70a3164393438646135656231636633 31
66633063616537303238636434306231613161336436386365376239326637333037303466343430
30353738623161386 10a6134333364393765616664393130373238393532636632613265383537 32
33636135643036646361653866656337306236653938383361313563623465393335626135633139
38303338636462643162376536343732353261396332633532643037663363633564306130623465
38353432366362333265663135366335666 6
```

b) Decrypt the helloworld.txt –

ansible-vault decrypt helloworld.txt

You can see the output as below –

```
[root@localhost vault-demos]# ansible-vault decrypt helloworld.txt
Vault password:
Decryption successful
[root@localhost vault-demos]#
```

c) Now again execute the cat command to see, if the file is decrypted –

cat helloworld.txt

The below output shows that the file is decrypted successfully

```
[root@localhost vault-demos]# cat helloworld.txt
Hi, I am demonstrating the ansible-vault encrypt command.
[root@localhost vault-demos]#
```

Note: There is no restriction on deleting the file which is encrypted. You can remove them using rm Unix command. It will not prompt for password.

Sub Activity 2: How to run the encrypted playbook?

Step 1	In Sub Activity 1, we have already encrypted on playbook. Let us execute the same playbook in this sub activity.

	Execute the ansible-playbook command on **playbook.yml** file and see if it executes –
	
	ansible-playbook playbook.yml
	
	You encounter the error as shown in the below image –
	
	`[root@localhost vault-demos]# ansible-playbook playbook.yml`
	`ERROR! Attempting to decrypt but no vault secrets found`
	`[root@localhost vault-demos]#`

Step 2	You can use **--ask-vault-pass** option which makes **ansible-playbook** command to prompt for vault-password
	
	ansible-playbook playbook.yml --ask-vault-pass
	
	You can see the playbook execution status as shown in the below image –
	
	`[root@localhost vault-demos]# ansible-playbook playbook.yml --ask-vault-pass`
	`Vault password:`
	
	`PLAY [target] ****************************************************************`
	
	`TASK [Create a file with some content] ****************************************`
	`changed: [192.168.10.129]`
	
	`PLAY RECAP ********************************************************************`
	`192.168.10.129 : ok=1 changed=1 unreachable=0 failed=0`
	`skipped=0 rescued=0 ignored=0`
	
	`[root@localhost vault-demos]#`

Step 3	Let us modify the **playbook.yml** file by editing the content of the file to be created on worker node represented by the group called target from “Hi, Hello, How are you?” to “Hi, Hello, How do you do?”. You can do this with ansible-vault edit command as shown below –
	
	playbook.yml file before modification –
	
	`---`
	`- hosts: target`
	`become: yes`
	`gather_facts: false`
	
	`tasks:`
	`- name: Create a file with some content`
	`copy: "dest=/tmp/file1.txt content='Hi, Hello, How are you?'"`
	
	playbook.yml file after modification –
	
	`---`
	`- hosts: target`
	`become: yes`
	`gather_facts: false`
	
	`tasks:`
	`- name: Create a file with some content`
	`copy: "dest=/tmp/file1.txt content='Hi, Hello, How do you do?'"`

Now execute the **ansible-playbook** command on **playbook.yml** file with -v option (verbose option which will display what is happened in detail.

ansible-playbook playbook.yml -v --ask-vault-pass

```
[root@localhost vault-demos]# ansible-playbook playbook.yml -v --ask-vault-pass
Using /etc/ansible/ansible.cfg as config file
Vault password:

PLAY [target] ******************************************************************************************************

TASK [Create a file with some content] *****************************************************************************
changed: [192.168.10.129] => {"ansible_facts": {"discovered_interpreter_python": "/usr/bin/python"}, "changed": true, "ch
ecksum": "8bc07b187c09ce0f11d6f4a0b3ad3901d6048168", "dest": "/tmp/file1.txt", "gid": 0, "group": "root", "md5sum": "d589
3e6cb8a496059a9ad3ad9902b54c", "mode": "0644", "owner": "root", "secontext": "unconfined_u:object_r:admin_home_t:s0", "si
ze": 25, "src": "/root/.ansible/tmp/ansible-tmp-1597772516.49-6376-188865956445701/source", "state": "file", "uid": 0}

PLAY RECAP *********************************************************************************************************
192.168.10.129             : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

This should not happen if the playbook is protected. We need to hide the task execution details to be shown even in verbose mode. To achieve this, we need to use attribute “no_log: true” in the playbook. Refer the **playbook.yml** file after modification.

```
---
- hosts: target
  become: yes
  gather_facts: false

  tasks:
  - name: Create a file with some content
    copy: "dest=/tmp/file1.txt content='Hi, Hello, How do you do?'"
    no_log: true
```

Delete the **/tmp/file1.txt** on worker node represented by **target** group.

Execute the below command and see the difference –

ansible-playbook playbook.yml -v --ask-vault-pass

```
[root@localhost vault-demos]# ansible-playbook playbook.yml -v --ask-vault-pass
Using /etc/ansible/ansible.cfg as config file
Vault password:

PLAY [target] ******************************************************************************************************

TASK [Create a file with some content] *****************************************************************************
changed: [192.168.10.129] => {"censored": "the output has been hidden due to the fact that 'no_log: true' was specified f
or this result", "changed": true}

PLAY RECAP *********************************************************************************************************
192.168.10.129             : ok=1    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@localhost vault-demos]#
```

Note: You can achieve the same, by setting attribute **no_logs = true** in **/etc/ansible/ansible.cfg** file only. But that will be general settings for all the tasks of all the playbooks.

Step 4

You can store the password in a file and give the path of the password file using the **--vault-password-file** option.

Create **pass-file** in your current working directory and store your password in it.

```
[root@localhost vault-demos]# echo '********' > pass-file
[root@localhost vault-demos]#
```

Give your vault password in this place

Execute the below command which will pick-up the password from pass-file for executing the **playbook.yml** –

ansible-playbook playbook.yml --vault-password-file=pass-file

Note: Name of the password file need not be **pass-file** only. It can be anything of your choice.

```
[root@localhost vault-demos]# ansible-playbook playbook.yml --vault-password-file=pass-file

PLAY [target] ******************************************************************************

TASK [Create a file with some content] *****************************************************
changed: [192.168.10.129]

PLAY RECAP *********************************************************************************
192.168.10.129             : ok=1    changed=1    unreachable=0    failed=0    skipped=0
                             rescued=0    ignored=0
[root@localhost vault-demos]#
```

Step 5

Set the attribute **vault_password_file** value to the path of **pass-file** (vault password file you have created in step 4). This will make all the encrypted playbooks will execute without prompting for the vault password.

To do so, execute **vi /etc/ansible/ansible.cfg** → Search for the attribute vault_password_file → once found press '**i**' to change to insert mode → give the path of the password file as a value of that attribute. → press '**Esc**' key → press '**:wq**' to make the changes reflect.

Now execute –

ansible-playbook playbook.yml

The above command will execute the playbook without prompting for the password even if it is encrypted. You can notice the same in the below image –

```
[root@localhost vault-demos]# ansible-playbook playbook.yml

PLAY [target] ******************************************************

TASK [Create a file with some content] *****************************
ok: [192.168.10.129]

PLAY RECAP *********************************************************
192.168.10.129             : ok=1    changed=0    unreachable=0
                             failed=0    skipped=0    rescued=0
                             ignored=0
[root@localhost vault-demos]#
```

Sub Activity 3: How to use encrypt_string in ansible-vault command?

Step 1	Sometimes, rather than encrypting complete playbook, we may need to encrypt only sensitive data (like passwords). To do so we may have to make use of encrypt_string in ansible-vault command.
	
	Create another playbook (may be named as **playbook1.yml**) with the below YAML code –
	
	`---`
	`- hosts: target`
	`  vars:`
	`  message: "This is my secret"`
	
	`  tasks:`
	`  - name: Output message`
	`  shell: echo {{ message }} >> /tmp/file3.txt`
	
	Here rather than encrypting the entire **playbook1.yml** file, I would like to encrypt only the value of the variable with the name **message** which is **"This is my secret"**

Step 2	Encrypt the string **"This is my secret"** and name it as **message** using **ansible-vault encrypt_string** command as shown in the below image –
	
	ansible-vault encrypt_string "This is my secret" --name message
	
	`[root@localhost vault-demos]# ansible-vault encrypt_string "This is my secret" --name message`
	`New Vault password:`
	`Confirm New Vault password:`
	`message: !vault |`
	`  $ANSIBLE_VAULT;1.1;AES256`
	`  36336138373661653931353139396365313638356666636366643830633131336138363430343635`
	`  34383731616663303232623331333039643932373630643 30a3461633733343231383565343239 33`
	`  38633132656133306636623030653930326663363266636434643732663838373061396561346430`
	`  37343335353834323 40a38346263393136376465353932326661363133633632646137323564343 6`
	`  64656365306637613432366166306435393334336430326463653738366635356534`
	`Encryption successful`
	`[root@localhost vault-demos]#`

Step 3	Copy the encrypted value of the variable message and paste it into playbook.yml file.
	
	playbook.yml file before replacing the value of the variable **message** with encrypted text –
	
	`---`
	`- hosts: target`
	`  vars:`
	`  message: "This is my secret"`
	
	`  tasks:`
	`  - name: Output message`
	`  shell: echo {{ message }} >> /tmp/file3.txt`

	playbook.yml file after replacing the value of the variable **message** with encrypted text –
	
	`---`
	`- hosts: target`
	`  vars:`
	`  message: !vault |`
	`  $ANSIBLE_VAULT;1.1;AES256`
	`  36336138373661653931353139396365313638356666636366643830633131336138363430343635`
	`  34383731616663303232623331333039643932373630643330a346163373334323138356534323933`
	`  38633132656133306636623030653930326663363266636434643732663838373061396561346430`
	`  37343335353834323340a383462633931363764653539323266613631336336326461373235643436`
	`  64656365306637613432366166306435393334336430326463653738366635356534`
	`  tasks:`
	`  - name: Output message`
	`  shell: echo {{ message }} >> /tmp/file3.txt`

Step 4	Execute the **ansible-playbook** command on **playbook1.yml** file, you can find the playbook is executed successfully.
	
	ansible-playbook playbook1.yml --ask-vault-pass
	
	`[root@localhost vault-demos]# ansible-playbook playbook1.yml --ask-vault-pass`
	`Vault password:`
	
	`PLAY [target] ********************************************************************`
	
	`TASK [Gathering Facts] ***********************************************************`
	`ok: [192.168.10.129]`
	
	`TASK [Output message] ************************************************************`
	`changed: [192.168.10.129]`
	
	`PLAY RECAP ***********************************************************************`
	`192.168.10.129 : ok=2 changed=1 unreachable=0 failed=0`
	`skipped=0 rescued=0 ignored=0`
	
	`[root@localhost vault-demos]#`

Step 5	You may go to worker node mentioned in the target group and see if **/tmp/file3.txt** is created. If created check if the content is same as "**This is my secret**"
	
	`[root@localhost ~]# cat /tmp/file3.txt`
	`This is my secret`
	`[root@localhost ~]#`